An Energy-Efficient Buffer Management Scheme Based on Data Integrity and Multivariate Data Reduction for Wireless Sensor Networks

Hassan Al-Mahdi [*], **Dina Hamed** [**], **Fayza Nada** [**], **Yasser Fouad** [**]

[*] *Department of Computer Science, Suez Canal University, Egypt Ismailia 41522 (e-mail: drhassanwesf@ci.suez.edu.eg).*
[**] *Suez University, Suez, Egypt (e-mail: (dina.hamed; f.nada; yasser.ramadan)@suezuni.edu.eg)*

Abstract: Buffer management and data reduction in wireless sensor networks (WSNs) are critical wherever buffer overflow and number of transmissions cause power waste and data loss. To improve energy consumption, this paper presents an Energy-Efficient Buffer Management based on data integrity and multivariate data reduction (EEBM-IMR) scheme. To save the buffer space, EEBM-IMR classifies the data measured by sensors, based on its integrity, into *malicious* or *verified* packets. To reduce data transmissions, a multivariate data reduction scheme is introduced based on a binary tree data structure. The efficiency of EEBM-IMR is evaluated in terms of transmission ratio, dropping probability and throughput using a real world dataset of fifty sensors. The experimental results show that, as the number of sensors increases, EEBM-IMR saves energy and outperforms some existing models in previous studies.

Keywords: Wireless sensor networks, data integrity, buffer management, IoT, data reduction, power consumption.

1. INTRODUCTION

Sensors, whether deployed in Wireless Sensor Networks (WSNs) or Internet of Things (IoT), are embedded within objects to sense and collect physical phenomenon from the world. The aggregated data are locally filtered and stored in a storage module called buffer which data are then conveyed over a network for storing, analyzing and processing in powerful fog and cloud servers (Adryan et al., 2017). WSNs are made up of two parts: sensor and sink nodes. Each set of sensors forms a cluster with a cluster head (CH) responsible for collecting data from the cluster members (CMs) and sending it to a sink node. The later forward data to a base station (BS) using either one hope or multi-hops fashion along different routes from the sender to receiver nodes (Kavitha and Suseendran, 2019).

Since the sensor nodes have very limited battery life (i.e., limited energy), their batteries must be replaced from time to time. On the other hand, most WSNs are installed in disaster, wild, hostile, battlefield, or underwater environments which makes changing battery very difficult. Since sensor nodes can be act as source nodes, routers and processing units, the energy represents a bottleneck in the WSN. Due to the massive number of sensor nodes, the data at CMs, CHs and sink nodes becomes very vast which resulted in buffer overflow and more packets get blocked or dropped. As the buffers get full, the number of retransmissions increases. As a result, sensor nodes lose significant amounts of energy during traffic transmissions.

Buffer management (BM) and data reduction are considered one of the most critical issues that pretty much affect the performance of WSNs. The efficient BM scheme should decrease the communication overhead and increase both the battery life (i.e., safe the power consumption) and the overall throughput of WSNs. Many buffer management schemes and data reductions have been proposed in the literature.

A Packet Priority Heterogeneous Queue (PPHQ) buffer management scheme introduced in (Rabileh et al., 2018) to minimize the important packets loss. The performance of PPHQ is evaluated in terms of packet delivery ratio, throughput, and end-to-end delay. On the other hand, this scheme ignores the malicious packets and the number of transmissions between the CHs and sink node. In (Jang et al., 2019), a new buffer thresholds-based model was implemented to improve the energy efficiency at cluster heads. The results were assessed using a mathematical and simulation model. However, the authors neglected to handle the redundant data at the buffers and the overhead of traffic transmissions between sensors and CHs.

In (Lodhi et al., 2020), a buffer management scheme for improving network efficiency was proposed. The authors higher the packets delivery by removing the packet bottleneck problem. However, their evaluation did not include discard rate of malicious packets and transmission ratio. Authors in (Jayarajan et al., 2020) proposed an Energy-Aware Buffer Management (EABM) routing protocol. They assume that WSN load can be distributed equally through increasing the number of clusters and decreasing the cluster size near the sink. Furthermore, EABM routing protocol satisfied higher lifetime, throughput, and lower packets drop when compared to LEACH routing protocol.

In (Shwe et al., 2010), a multilayer WSN is discussed with power efficient buffer management policy. The storage space

[⋆] Sponsor and financial support acknowledgment goes here. Paper titles should be written in uppercase and lowercase letters, not all uppercase.

is shared efficiently in each sensor node such that maximum throughput and minimum recovery cost of packets lost are achieved. The proposed model reduces packets loss through dividing the network topology into three layers such that each layer is interested in different data from the surrounding environment. This scheme, on the other hand, ignored malicious packets, which raise the risk of buffer overflow. Authors in (Ghazi et al., 2018) presented a prioritized policy for controlling congestion in WSN. This strategy eliminates congestion and fixes the problem when it does occur by returning the network to a consistent state. In view of the obtained results, packets loss and power consumption have been improved. However, they fail to satisfy the varying Quality-of-Service (QoS) requirements of heterogeneous applications.

A Priority based Adaptive Scheduling Algorithm (PASA) for IoT sensor systems was proposed in (Kavitha and Suseendran, 2019) . The suggested scheduling put into consideration the demands of heterogeneous applications. Depending on the degree of traffic priority, remaining buffer size, and necessary transmission energy, each sensor node is allocated collision-free time slots. In view of experimental results, the performance of PASA has been found to be improved in terms of throughput and residual energy of nodes.

In WSNs, data reduction is generally used to save energy by reducing the transmission of sensor readings across the network. Several studies on WSN data reduction have been released. Reducing and cleaning redundant data will save energy and resources, which reduces connectivity costs. In (Idrees et al., 2020), the authors presented Data Reduction and Cleaning Approach (DaReCA) scheme to save energy of WSNs of IoT. Two levels of data reduction and cleaning are applied. In the former, a cleaning algorithm is adopted by the sensor node for removing redundant data before sending to the aggregator. Next, divide and conquer method is implemented at the aggregator for merging and reducing similar data sets received from the sensor nodes before sending to the sink node. The obtained results showed better performance regarding energy consumption at both the wireless sensors and the aggregator.

A prediction model based on Extended Cosine Regression (ECR) was proposed in (Jain and Kumar, 2020). A high accuracy and minimized-energy consumption with successful predictions was introduced. Moreover, to minimize the cumulative errors resulted from repeated predictions and to synchronize the predicted data, ECR applied a two-vector model. The simulation results indicated that the introduced technique achieves better prediction accuracy energy preservations.

This paper aims to introduce an Energy-Efficient Buffer Management based on data integrity and multivariate data reduction (EEBM-IMR). The main contributions of the proposed EEBM-IMR scheme are summarized as follows:

(1) Buffer integrity-based scheduling policy is developed. To our knowledge, most of proposed models in Literature ignored the data integrity during the buffer scheduling process. In fact, integrity insures that the data is not tampered by unauthorized individuals (Ertürk et al., 2019). In EEBM-IMR, data packets integrity will be taken into consideration to safe the buffer size and to prolong the battery life. Without data integrity, a cluster head is not aware of data changes (Burhan et al., 2018). As a result, the buffer space can be filled by several malicious packets, which leads to buffer overflow and drastically reduces the effective network throughput.

(2) A multivariate data reduction algorithm based on a binary tree is developed to save the energy consumption resulting from radio transmission. To comply with that, EEBM-IMR should be able to reduce the number of transmissions between cluster heads and sink node.

(3) The proposed EEBM-IMR increases the overall throughput using data integrity. In doing so, the measured data are filtered at the buffer of cluster heads to remove all malicious packets.

Actually, there are two differences between our multivariate data reduction algorithm introduced in EEBM-IMR scheme and those in (Xu and Zhang, 2017; Arbi et al., 2017; Mohamed et al., 2018). First, these algorithms are independently applied to each sensor (i.e., single variable), while we consider dependent sensors (multivariate data). The second is that they are appropriate for clusters with few sensors, but our proposed EEBM-IMR scheme is suitable for clusters with a large number of sensors.

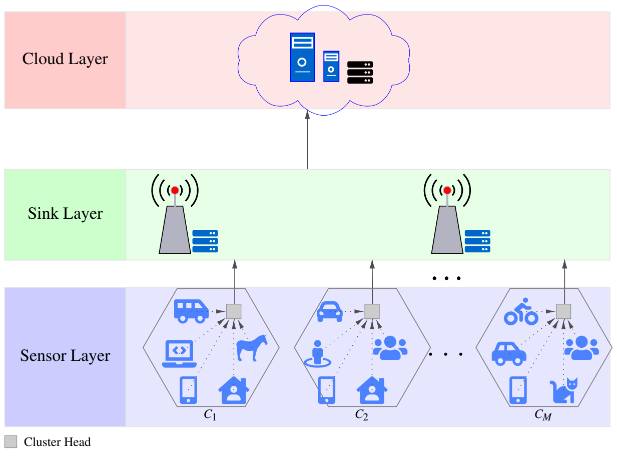

Fig. 1. Conceptual structure of the EEBM-IMR scheme.

2. NETWORK MODEL

As shown in Figure 1, the WSN that we are considering in this paper, consists of a massive number of tiny sensor nodes deployed in a particular geographic area. The sensor nodes are grouped in M clusters. Each cluster C_i $(i = 1,2,3,\ldots,M)$ has a coordinator called cluster head CH_i and a set of sensor nodes called cluster members (CMs). CHs expend more resources than CMs and are in charge of aggregating data from CMs and sending it to the Sink node (BS) for further analysis (Jain and Reddy, 2014). CMs within each a cluster can sense data like temperature, pressure, light, vibration, sound, radiation and humidity. Many clustering methods has been proposed in literature (Ullah and Youn, 2019, 2020).

The LEACH is one of the most popular clustering protocol introduced in literature (Heinzelman et al., 2002). In addition, enormous clustering techniques are introduced in literature to select cluster head nodes based on some criteria like weighting network nodes, battery mathematical cluster model, K-step overlap, maximum residual energy and energy harvesting (Sah and Amgoth, 2020; Zainalie and Yaghmaee, 2008). Let $\phi = 1,2,3,\ldots,n$ and $\theta = 1,2,3$ denoting CM type and mode respectively, where n denotes the number of CM types within

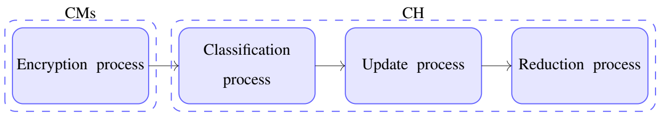

Fig. 2. The EEBM-IMR components.

cluster C_i. A CM mode can be either awake ($\theta = 0$), sleep ($\theta = 1$) or idle ($\theta = 2$). Idle mode means a CM no longer working or maybe hacked by attackers.

Let CM_j^i $(j = 1, 2, 3, \ldots, N)$ denotes the j^{th} CM within the i^{th} cluster C_i, where N denotes the number of CMs in the cluster C_i. The CMs of the cluster C_i take the responsibility to sense and transmit data to its CH_i. Let $ID_1^{i,j}$ and ID_2^i represent the identifiers of CM_j^i and CH_i respectively. The values of $ID_1^{i,j}$ are initially stored at CH_i. The CH_i gathers data from its CMs and forwards to the sink node for further analysis and decision making via one-hop or multi-hop. During transmission from a certain CH_i to sink node, packets may pass through intermediate CH nodes. The selection of CH nodes across the path is based on the routing table maintained in each CH.

We assume that arriving data packets at CH_i are classified into two classes: *class-1* and *class-2*. *Class-1* arrivals denote the data packets originated from CMs of the cluster C_i. *Class-2* arrivals denote the data packets originated from the CH_k, $i \neq k$. Each CH_i has a finite buffer with size B for buffering *class-1* and *class-2* arriving packets. Sink node, on the other hand, has a finite buffer with size K for buffering data packets arriving from the different clusters heads.

Due to the limited capacity of the buffer size at CHs, data loss at these CHs will occur during the transmission of packets from CMs to sink node. This leads to energy consumption, decreased throughput, and increased delay. The proposed EEBM-IMR, as shown in Figure 2, is designed to include four interactive processes for serving a packet. Namely, encryption, classification, update and reduction processes. EEBM-IMR uses a lightweight encryption algorithm in the encryption phase to encrypt data at CMs. During the classification phase, the arriving packets are filtered and queued at CHs. The previous measures for CMs are updated in the update phase, based on the output of the second phase. In reduction phase, the updated data packets are reduced. The implementation of EEBM-IMR generally consists of two steps:

- Step 1: The data encryption process is carried out of the CMs.
- Step 2: The processes of classification, updating and reduction are carried out of CHs.

3. EEBM-IMR DESCRIPTION

As shown in Figure 3, the total buffer size B of each CH_i is divided into number of virtual buffers. Namely, Local Buffer (LB), Forward Buffer (FB), History Buffer (HB), Reduction Buffer (RB). The size of LB, FB, HB and RB are N_1, N_2, N_3 and N_4 respectively, where $N_1 + N_2 + N_3 + N_4 \leq B$. The LB buffer of CH_i stores data packets measured by CMs within C_i at a certain time.

The HB stores history measured data packets of the CMs within C_i, where each CM has an entry in the HB buffer. On the other hand, RB temporally stores the reduced data of packets residing in the HB. Finally, FB receives packets coming from various CHs. Moreover, each CH_i is equipped with an Integrity Check Module (ICM). The ICM is responsible for classifying *class-1* arriving data packets as *verified* or *malicious* packet.

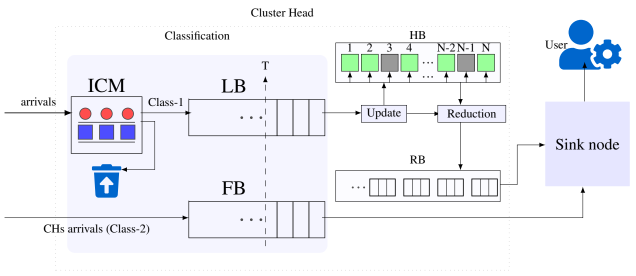

Fig. 3. Conceptual structure of the EEBM-IMR scheme.

3.1 *Encryption Process*

Encryption process is first phase in EEBM-IMR, where each CM encrypt its measured data using similar method introduced in (Elshrkawey and Al-Mahdi, 2021). The encryption process consists of two parts. The first part is implemented in CH, while the second part is very light and implemented in CM. Authors (Elshrkawey and Al-Mahdi, 2021) illustrate that, the very light encryption at the CM-side has little impact on the power consumption, specially when the data to be encrypted at CM-side is very small.

Let K_{prv}^i and K_{pub}^i are initial private and public keys respectively at the cluster CH_i. These keys are generated according to the Elliptic Curve Cryptography (ECC) (Boudia et al., 2017). The value of K_{prv}^i is initially stored at the cluster head CH_i and its CM_j^i $(j = 1, 2, 3, \ldots, N)$. The CH_i generates the two points $S_{CH}^i = \left(S_x^i, S_y^i\right)$ and $A_{CH}^i = \left(A_x^i, A_y^i\right)$ based on the public key K_{pub}^i as follows.

$$S_{CH}^i = \text{rand}\,(1, n-1)\, K_{pub}^i = \left(S_x^i, S_y^i\right) \tag{1}$$

and

$$A_{CH}^i = \text{rand}\,(1, n-1)\, K_{pub}^i = \left(A_x^i, A_y^i\right) \tag{2}$$

For propose of security, the pairs $\left(S_x^i, S_y^i\right)$ and $\left(A_x^i, A_y^i\right)$ are changed from session to session. For each session, the cluster head CH_i sends the two points $\left(S_x^i, S_y^i\right)$ and $\left(A_x^i, A_y^i\right)$ to its CMs over a secure channel. Let K_{data}^i, K_{MAC}^i and K_{ID}^i are session keys used to cipher/decipher data, digital signature and identifier of each CM_j^i, respectively. The cluster head CH_i and its cluster members CM_j^i use S_{CH}^i and A_{CH}^i to independently generate the session keys K_{data}^i, K_{MAC}^i and K_{ID}^i using the well-known key derivation function $KFD(x, y)$ Barker et al. (2018) as follows.

$$K_{data}^i = KFD\left(\left(S_x^i, A_x^i\right), K_{prv}^i\right), \tag{3}$$

$$K_{MAC}^i = KFD\left(\left(S_y^i, A_y^i\right), K_{prv}^i\right) \tag{4}$$

and

$$K_{ID}^i = KFD\left(\left(K_{data}^i, K_{MAC}^i\right), K_{prv}^i\right) \tag{5}$$

Before sending the data packet ($dPkt$) from the CM_j^i to its cluster head CH_i, the CM encrypts the $dPkt$ through 32 stages of the Unbalanced Feistel network. Feistel method was invented by Horst Feistel, where the Feistel function, F, takes the packet $dPkt$ and the key K_{data}^i as inputs and the output is a cipher packet ($cPkt$) as follows:

$$cPkt = F\left(dPkt, K_{data}^i\right)$$

Next, CM_j^i generates its signature T_j^i as follows:

$$T_j^i = cPkt \otimes ID_1^{i,j} \otimes K_{MAC}^i \tag{6}$$

The final encrypted message M_i^j that will be sent from CM_j^i to its CH_i is given as:

$$M_i^j = E_{K_{ID}^i}\left\{cPkt \parallel T_j^i \parallel ID_1^{i,j} \parallel \phi\right\} \tag{7}$$

Assume that the attacker succeeded to obtain the encrypted data M_i^j of the cluster member CM_j^i. He tries to obtain the plain text and signature T_j^i of the cluster member CM_j^i in two steps as follows.

(1) Step 1: He tries to determine the value of K_{ID}^j which used to decrypt the encrypted data M_j^i. To do this, He needs all the possible values of the keys K_{data}^i, $ID_1^{i,j}$ and K_{ID}^i. The length of each one of these keys are 32 bits (i.e., 4 bytes). As a result, the number of trials needed for this step is given as $V_1 = 2^{96}$.
(2) Step 2: Assuming that, the attacker succeeded in step 1. In such case, He tries to determine the signature T_j^i. The tag signature of each M_j^i was calculated based on the values of $cMsg$, $ID_1^{i,j}$ and K_{MAC}^i. As a result, the number of trials needed for this step is given as $V_2 = 2^{64}$.

Regarding to the two steps, the probability of hacking is given as follows:

$$P_h = \frac{1}{V_1}\frac{1}{V_2}$$

3.2 Data classification

In this section, the classification and scheduling policy of measured data are illustrated. As shown in Figure 3, classification is the second phase of EEBM-IMR that applied at CH_i to treat the encrypted *class-1* incoming messages originating from CMs. Upon the CH_i of cluster C_i receives the encrypted message M_i^j from CM_j, the ICM module starts to execute the classification process as follows.

The ICM module decrypts the M_i^j message using K_{ID}^i to extract the encrypted data packet $cPkt$, the tag signature T_j^i and the cluster member identifier $ID_1^{i,j}$. It then recalculates the tag signature T_j^{new} using equation (6). Based on the comparison between the received tag signature T_j^i and the new calculated tag signature T_j^{new}, the data packet is categorized into either *verified* or *malicious* packet. *Malicious* packet is dropped and *verified* is treated based on the value of the threshold T as follows.

(1) If LB queue is not full then the *verified* packet is accepted.
(2) If LB queue is full and the number of packets in FB queue is less than the threshold T then one free space is borrowed from FB and *verified* packet is accepted. The FB queue size is decremented by one (i.e., $N_2 = N_2 - 1$).
(3) If LB and FB are full then the *verified* packet is dropped.

The ICM reconstructs the *verified* data packet as $\varpi_j = \{data, ID, \phi\}$ and stores it at the tail of the LB queue, where $ID = ID_1^{i,j}$. Dropping the *malicious* packets will maximize the useful data, save the buffer space and maximize the overall throughput by minimizing the number of retransmitted packets. The execution steps of the data classification are illustrated in algorithm 1. On the other hand, the *class-2* arriving messages are accepted if the size of FB is less than $max\{N_2, T\}$ and dropped otherwise.

Algorithm 1 Classification algorithm at CH_i

```
1:  Input: $M_j$
2:  Output: $\varpi_j$
3:  Extract the values of $cPkt$, $T_j^i$, $ID_1^{i,j}$ and $\phi$ from the received $M_j$
4:  if $T_j^{new} = T_j^i$ then
5:      Mark cPkt as verified packet
6:      Extract the measured value $m_j$ from the received cPkt
7:      $data \leftarrow m_j$
8:      $ID \leftarrow ID_1^{i,j}$
9:      $\varpi_j \leftarrow \{data, ID, \phi\}$
10:     if $LB.length < N_1$ then
11:         Queue in $\varpi_j$ in the LB queue
12:     else
13:         if $LB.length = N_1$ and $FB.length \le T$ then
14:             Borrow one free space from the FB queue
15:             $LB.size = LB.size + 1$
16:             Queue in $\varpi_j$ in LB
17:         else
18:             Drop $\varpi_j$
19:         end if
20:     end if
21: else
22:     cPkt is marked as malicious packet
23:     Drop the malicious packet
24: end if
```

3.3 Data Update Process

This is an ongoing process which updates the history measures of the whole CMs for the reduction phase. The data update process is the third phase in EEBM-IMR scheme where by the previous measures contained in the HB buffer are updated based on the current measures ϖ_j residing in the LB buffer.

The HB buffer consists of $N_3 = N$ storage elements, where each element $j = 1, 2, 3, \ldots, N$, is used to store the history measured value h_j of the CM j. In other word, the sequence $W = \{\varpi_1, \varpi_2, \varpi_3, \ldots\}$ constructed at LB buffer is used to update the values of the sequence $H = \{h_1, h_2, h_3, \ldots, h_N\}$ at HB buffer, where $W \le H$. Regarding to the stored measured values in LB and HB buffers, the relationship between the sequences H and W can be either *Normal* or *Abnormal* case as follows.

(1) *Normal* case. All elements of the sequence H will be updated (i.e., $H = W$). This is achieved only if all current measures of CMs are marked as a *verified* packets and accepted at the LB buffer.
(2) *Abnormal* case. In this case some elements in H have no equivalent in W (i.e., $H > W$). This case takes place in either one of the following three situations:
   - The ICM module classifies some measured data packets as *malicious* and get dropped.
   - The ICM module classifies some measured data packets as *verified* and get dropped because there is no free buffer space in LB.
   - Some sensors may did not send data due to battery life or they are no longer working.

For clarity, an example of six sensors in Figure 4 illustrates the possible situations of the *Normal* and *Abnormal* cases. In normal case, all sensors send their measured data and marked as *verified* packets and accepted during the classification process. In abnormal case 1, packets of sensors 3 and 6 are marked as *malicious* by the ICM module and get dropped while the remaining data packets are accepted. Finally, in abnormal case 2, sensor 2 does not work while the data packet of sensor 4 is marked as *verified* via ICM module but get dropped due to the buffer space. The remaining data packets are accepted.

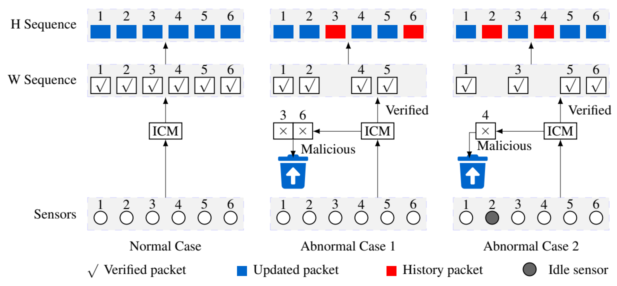

Fig. 4. An example of classification and update processes.

3.4 Data Reduction

In this process, instead of transmitting all data packets residing in the HB queue to the sink node, the EEBM-IMR scheme decreases the number of packets transmitted via the execution of the data reduction process based on binary tree data structure as depicted in algorithm 2 as follows.

The data packets $h_j, (j = 1,2,3,\dots,N)$ residing in the HB buffer are arranged into sequences S_k, $1 \le k \le n$, based on the sensor type ϕ, where S_k contains all the data packets originated from sensors of the type $\phi = k$ and n denoting the number of sensor types within the cluster C_i. For each sequence S_k, we construct a binary tree data structure as follows. The first parent (root) of the binary tree is the leftmost element in the sequence S_k (i.e. $r_0 = S_k[0]$). For the remaining elements of the sequence S_k, construct the list L_1 such that

$$|r_0 - S_k[j]| < \varepsilon_k, j = 1,3,\dots,S_k.length - 1 \tag{8}$$

and construct the list R_1 such that

$$|r_0 - S_k[j]| \ge \varepsilon_k, j = 1,3,\dots,S_k.length - 1 \tag{9}$$

where the tolerance ε_k is non-negative real number for sensors of type k.

The two lists L_1 and R_1 represent the children of the parent r_0. Repeat this process for the list R_1 to generate the parent r_1 and the two children L_2 and R_2. The child R_1 is replaced with the parent r_1. The two lists L_2 and R_2 represent the children of the parent r_1. This process continue until we reach the parent r_w $(w = 0,1,2,\dots,E)$ with the empty child list R_{w+1}, where E denotes the number of parent nodes in the generated binary tree. The continent of each parent node r_w is reformulated as follows:

$$r_w = \{data, \phi = k, ID\} \tag{10}$$

where $ID = ID_2^i$ is the identifier of the CH_i. By the end of binary tree construction, the parents $r_0, r_1, r_2, \dots, r_E$ are moved one by one into the RB queue. The RB queue entries are then transmitted to the sink node using FIFO policy. Based on the value of E, we have three different cases:

- Best case ($E = 1$): In such case, the elements of the sequence S_k are reduced into one data packet.
- Moderate case ($1 < E \le \lfloor S_k.length/2 \rfloor$): In such case, the elements of the sequence S_k are reduced into more than one data packet.
- Worst case ($E = S_k.length$): In such case, the sequence S_k is not reduced and all its elements are moved into the RB queue.

Algorithm 2 Data Reduction Process at the CH_i

```
1:  Input: Sensor types $n$, updated history values $h_1, h_2, h3, \dots, h_N$
2:  Output: $r_0, r_1, r_2, \dots, r_E$
3:  for all $k \in n$ do
4:    for $j \in N$ do
5:      if HB.$h_j.\phi = k$ then
6:        Add $h_j$ to $S_k$
7:      end if
8:    end for
9:    Set $z = 0$
10:   while $S_k$ is not empty do
11:     $r_z.data = S_k[0].data$
12:     $r_z.\phi = S_k[0].\phi$
13:     $r_z.ID = ID_2^i$
14:     Queue in $r_z$ into RB queue
15:     remove $S_k[0]$
16:     for all $J \in S_k.length$ do
17:       if $|r_z - S_k[j]| < \varepsilon_k$ then
18:         Add $S_k[j]$ to $L_{z+1}$
19:         remove $S_k[j]$
20:       end if
21:     end for
22:     $R_{z+1} = S_k$
23:     $z = z + 1$
24:   end while
25: end for
```

For clarity, Figure 5 illustrates the construction of binary tree data structure for 16 sensors of type 2 (i.e. $\phi = 2$). These measures are arranged into sequences $S_2 = \{h_1, h_2, h_3, \dots, h_{16}\}$, where $h_j.\phi = 2$ for all $j = 1,2,3,\dots,16$. As shown in Figure 5-(a), the best case is achieved only if the number of parents in the generated binary tree is equal to one (i.e., $E = 1$). In this case $|h_1 - S_k[j]| < \varepsilon_k$ for $j = 2,2,3,\dots,16$). In the moderate case (b), the number of parents $E = 3$. On the other hand, As shown in Figure 5-(c), the worst case arises if the number of parents in the generated binary tree is equal to 16 (i.e., $E = 15$).

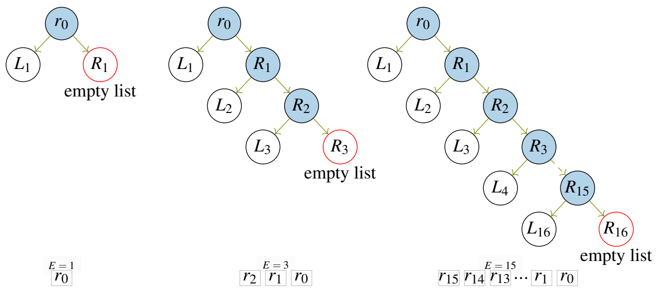

Fig. 5. An example showing how the entries of the HB queue are reduced based on the the tolerance ε_2 and sensors type $\phi = 2$.

3.5 Sink node operations

As aforementioned, the goal of the proposed EEBM-IMR is to lower the number of transmissions from the cluster heads to the sink node, and at the same time save the buffer space at the cluster heads and decrease the power consumption. We assume that the transmission from the CH_i to the sink node can be either *normal* mode or *reduction* mode.

In *normal* mode, the CH_i transmit data from the HB to the sink node without reduction while in *reduction* mode the CH_i transmits the measured data from the RB to the sink node after executing the reduction process. Initially, the CH_i transmits the measured data to the sink node in *normal* mode until make sure that all its CMs have history data stored at the sink node. Next, the CH_i switch to *reduction* mode and starts to transmit the data packets residing in the RB to sink node.

The sink node has a buffer space named Sink Buffer (SB) with enough size to host the data packets received from all clusters (i.e., each sensor has a history measured data stored at the sink node). Let H_j, $j = 1,2,3,\dots,M \times N$ denotes the history measured data of a sensor j. Upon arrival of a data packet r_k at the sink node, it stats to update the history data H_j based on the values $r_k.\varepsilon_k$, $r_k.ID$ and $r_k.\phi$ as shown in algorithm 3.

Algorithm 3 Sink (receiver) Algorithm

```
1: Input: Reduction value $r_k$
2: Output: Updated value $H_j$
3: for all $j \in M \times N$ do
4:     if $r_k.ID = H_j.ID$ AND $r_k.\phi = H_j.\phi$ then
5:         if $|r_k.data - H_j.data| < r_k.\varepsilon_k$ then
6:             $H_k.data = r_k.data$
7:         end if
8:     end if
9: end for
```

4. PERFORMANCE EVALUATION

This section aims to evaluate and validate the proposed EEBM-IMR scheme. Using the Python programming language, simulation is carried out because it is flexible and scalable.

We evaluate the performance of our EEBM-IMR scheme against the data reduction scheme (LEERT) in (Mohamed et al., 2019). LEERT, as reported in (Mohamed et al., 2019), applies the same prediction model independently on both sensors and sink nodes. In the LEERT scheme, on the other hand, data integrity is not taken into consideration.

The operational parameters, performance metrics and simulation results are discussed in this section. For fair investigations, the EEBM-IMR and LEERT schemes are simulated using the same real world Intel Lab dataset collected by the Intel Berkeley Research Lab (IBRL) (Madden, 2018). This dataset contains about 2.3 million measures collected from 54 sensors with identifiers range from 1-54. Each sensor can measure temperature, humidity, light and voltage. In this paper, we only concentrate on the temperature and humidity readings from IBRL dataset.

The sensors arranged into two clusters (i.e. $M = 2$) C_1 and C_2. The number of CMs N within each cluster ranges from 5 to 50 sensors. The maximum residual energy criteria is used to select cluster heads as in (Zainalie and Yaghmaee, 2008). The CMs and CHs within the network are deployed randomly in 450×450m^2 simulation area. All sensors are static, homogenous and with same battery resource. The sink node (SB) is operated by a permanent power supply and located outside of the network area. The simulations were made based on many parameters as shown in table 1. The proposed EEBM-IMR is evaluated in terms of the metrics: transmission ratio, dropping probability, waiting time and throughput.

Table 1. Simulation parameters.

Parameter	Value
Number of Sensor Nodes	5-50 sensors
Range of Transmission	30m
Dimensions of Work Area	450×450 m^2
Transmission Power	0.650, 0. 125 mw
Preliminary Energy	7.2 J
Simulation Time	10^6 msec.
Buffer size N	10,20,30,50 packets
Arriavl rate	vary
Hacking rate P_h	0.1, 0.2 packets

4.1 Transmission Ratio

In this subsection, we evaluate the proposed EEBM-IMR scheme in terms of the transmission ratio (TR) metric which given in (Zainalie and Yaghmaee, 2008) as $TR = \frac{d_{tr}}{d_{or}}$ where d_{tr} denotes the number of packets transmitted between cluster head and sink node, and d_{or} denote the size of original data aggregated at sink node. It is clear that, less TR means fewer data transfers between the cluster head and the sink node, and thus less energy consumption.

scenario 1: For Scenario 1, the number of CMs N is set to 30 and 50 sensors. The buffer size is fixed to 100 packets. For a fair comparison, the hacking rate P_h is set to 0. This ensures that the integrity check module (ICM) will not detect any *malicious* packets.

As in Figure 6, We assume that, all CMs from the same type and the simulation run is conducted based on the extracted temperature measurements over a period of 10 days from IBRL dataset. Figure 6 illustrates the transmission ratio, TR, versus tolerance ε for the EEBM-IMR and LEERT reduction schemes, where ε ranges from 0.1^oC to 4^oC.

Note that, ε in LEERT used as forecasting tolerance while ε in EEBM-IMR is used to construct the reduction binary tree data structure. The Figure indicates that when ε gets larger, TRs of the two schemes get smaller.This intuitively clear, because as ε gets larger, the CH will find more forecasted values in the case of LEERT scheme and the CH will find fewer elements in the binary tree in the case of EEBM-IMR scheme. Moreover, the proposed EEBM-IMR scheme outperforms LEERT when N is large. This implies that, EEBM-IMR is better than LEERT when WSN contain an enormous number of sensors.

Figure 7, on the other hand, demonstrates the TR of both schemes when $N \leq 10$. The Figure indicates that LEERT scheme achieve high data reduction than EEBM-IMR when a cluster has small number of sensors. However, problems arise with LEERT when trying to deploy it to support an application with high number of sensors.

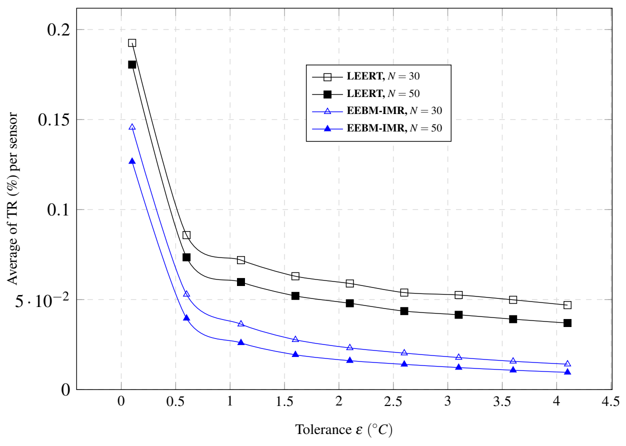

Fig. 6. Transmission ratio versus tolerance ε high number of sensors per cluster.

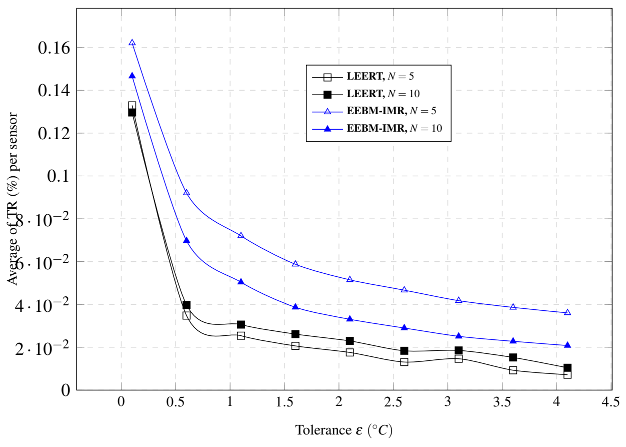

Fig. 7. Transmission ratio versus tolerance ε low number of sensors per cluster.

Scenario 2: In this scenario, we first examine the impact of dataset size on the cumulative data reduction. To do this, we will use $N = 40$ sensors, tolerance $\varepsilon = 0.1°C$ and number of cluster $M = 2$. Figure 8 shows the transmission ratio, TR versus the IBRL dataset samples range from 2000 to 14000 measures. The Figure indicates that the proposed EEBM-IMR outperforms the LEERT scheme.

Second, we examine the impact of CMs size (i.e., number of sensors) on the transmission ratio, TR. The simulation is carried out for schemes using 15,000 temperature measures from the IBRL dataset. Figure 9 illustrates that:

(1) For lower number of sensors within a cluster (i.e. $N < 10$), the TR decreases for both schemes. In addition, the LEERT outperforms our proposed scheme.
(2) For a higher number of sensors within a cluster (i.e. $N > 10$), our EEBM-IMR scheme performs well with more sensors pervasive in a cluster than the LEERT scheme. We have observed from Figure 9 that as $N > 10$, the LEERT increases TR while the EEBM-IMR decreases TR. This enhancement is attributable to the fact that in the reduction process, our proposed EEBM-IMR uses a multivariate dataset while LEERT independently applies its prediction model on each sensor.

.

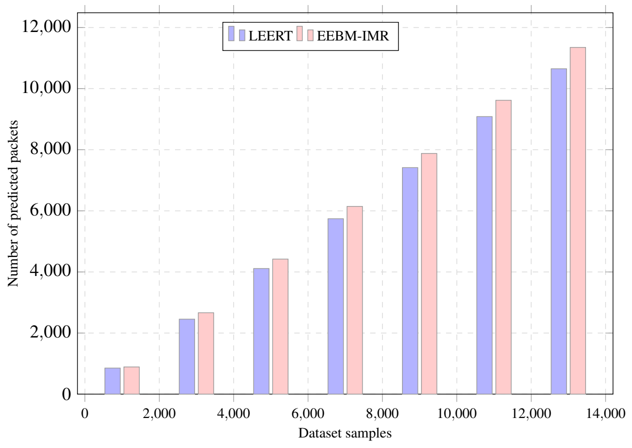

Fig. 8. Transmission ratio versus dataset samples.

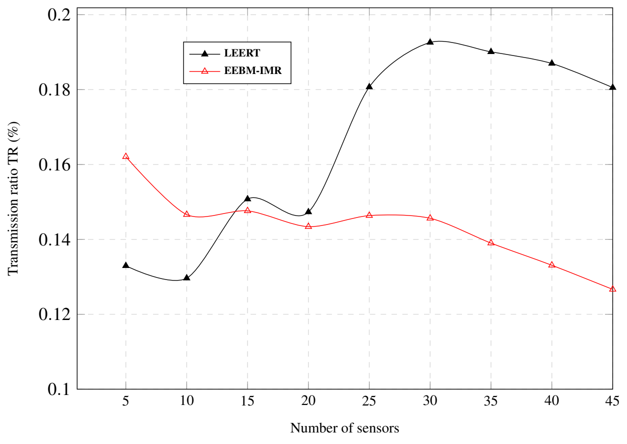

Fig. 9. Transmission ratio versus number of sensors

4.2 *Dropping Probability*

In this section, the proposed EEBM-IMR scheme is evaluated in terms of dropping probability. The buffer size is set to 50 packets. By turning off the ICM module, more *malicious* packets are accepted, leaving less room for *verified* packets. Furthermore, when ICM module is enabled, all *malicious* packets are rejected, making more room for *verified* packets.

Two simulation runs are used to investigate the effect of ICM module on dropping probability. We disable the ICM module for the first run and enable it in the second run with hacking rates of $Ph = 0.1$ and $Ph = 0.3$ packets/second. Figure 10 shows that, when ICM module is enabled the dropping probability of the verified packets decreases. Figure 11 illustrates the buffer occupancy when ICM module is disabled.

Figures 10 and 11 show that, as the dataset sample size expands, the number of *malicious* packets increases. The data reduction algorithms of EEBM-IMR and LEERT are negatively affected by this number.This is because these algorithms will run on false data as the buffer fills up with more malicious packets. As a result, total throughput will decrease, while power consumption at cluster heads will increase.

5. CONCLUSION

In this paper, an integrity based buffer management and multivariate data reduction scheme (EEBM-IMR) for WSN are

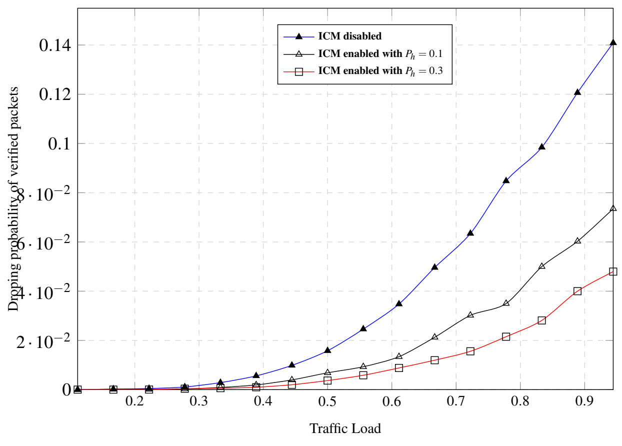

Fig. 10. Dropping probability versus traffic load.

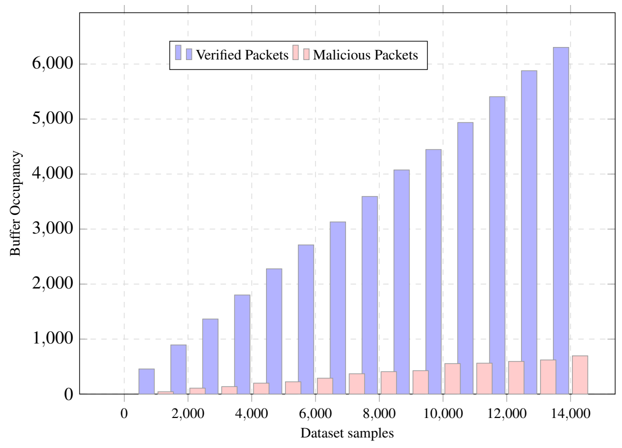

Fig. 11. Buffer occupancy versus dataset samples.

suggested. A lightweight encryption algorithm is introduced at both the sensors and cluster heads. The collected data at cluster heads are filtered using ICM module based on the data integrity. To minimize the number of transmissions between cluster heads and sink node, a multivariate data reduction algorithm is introduced. The proposed EEBM-IMR has two key goals: it saves buffer space by rejecting all tampered data and it decreases power consumption at cluster heads by using a multivariate data reduction scheme. The Python framework and a real-time dataset are used to test and validate the performance of EEBM-IMR against the LEERT data reduction algorithm. The simulation results show that our proposed EEBM-IMR outperforms LEERT data reduction algorithm in terms of transmission ratio, power consumption and number of sensors. Also, the dropping probability of untampered data was reduced. In addition, the EEBM-IMR is well-suited to applications involving a large number of sensors. In the future, it is intended to modify the proposed EEBM-IMR scheme to address the issue of excessive delay in the real-time applications and implement the buffer scheme based on a more general service time distribution.

REFERENCES

Adryan, B., Obermaier, D., and Fremantle, P. (2017). *The technical foundations of IoT*. Artech House.

Arbi, I.B., Derbel, F., and Strakosch, F. (2017). Forecasting methods to reduce energy consumption in wsn. In *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 1–6. IEEE.

Barker, E., Barker, E., Chen, L., and Davis, R. (2018). *Recommendation for key-derivation methods in key-establishment schemes*. US Department of Commerce, National Institute of Standards and Technology.

Boudia, O.R.M., Senouci, S.M., and Feham, M. (2017). Elliptic curve-based secure multidimensional aggregation for smart grid communications. *IEEE Sensors Journal*, 17(23), 7750–7757.

Burhan, M., Rehman, R.A., Khan, B., and Kim, B.S. (2018). Iot elements, layered architectures and security issues: A comprehensive survey. *Sensors*, 18(9), 2796.

Elshrkawey, M. and Al-Mahdi, H. (2021). Sda-sm: An efficient secure data aggregation scheme using separate mac across wireless sensor networks. *International Journal of Computers, Communications & Control*, 16(2).

Ertürk, M.A., Aydın, M.A., Büyükakkaşlar, M.T., and Evirgen, H. (2019). A survey on lorawan architecture, protocol and technologies. *Future Internet*, 11(10), 216.

Ghazi, M.U., Naqvi, S.S.H., Yamin, K., and Humayun, O. (2018). Congestion-aware routing algorithm based on traffic priority in wireless sensor networks. In *2018 15th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT (HONET-ICT)*, 112–116. IEEE.

Heinzelman, W.B., Chandrakasan, A.P., and Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on wireless communications*, 1(4), 660–670.

Idrees, A.K., Abou Jaoude, C., and Al-Qurabat, A.K.M. (2020). Data reduction and cleaning approach for energy-saving in wireless sensors networks of iot. In *2020 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)(50308)*, 1–6. IEEE.

Jain, A. and Reddy, B. (2014). Sink as cluster head: An energy efficient clustering method for wireless sensor networks. In *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)*, 1–6. IEEE.

Jain, K. and Kumar, A. (2020). An energy-efficient prediction model for data aggregation in sensor network. *Journal of Ambient Intelligence and Humanized Computing*, 11(11), 5205–5216.

Jang, Y., Shin, A., and Ryoo, I. (2019). Energy efficiency improvement based on optimal buffer thresholds model for wireless sensor devices. In *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, 182–187. IEEE.

Jayarajan, P., Kanagachidambaresan, G., Sundararajan, T., Sakthipandi, K., Maheswar, R., and Karthikeyan, A. (2020). An energy-aware buffer management (eabm) routing protocol for wsn. *The Journal of Supercomputing*, 76(6), 4543–4555.

Kavitha, K. and Suseendran, G. (2019). Priority based adaptive scheduling algorithm for iot sensor systems. In *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, 361–366. IEEE.

Lodhi, A.K., Rukmini, M., Abdulsattar, S., and Tabassum, S.Z. (2020). Performance improvement in wireless sensor networks by removing the packet drop from the node buffer. *Materials Today: Proceedings*, 26, 2226–2230.

Madden, S. (2018). Intel berkeley research lab data. *http://db.csail.mit.edu/labdata/labdata.html.*

Mohamed, M.F., Ahmed, M.A., and Nassar, H. (2019). Lightweight energy-efficient framework for sensor real-time communications. *IET Communications*, 13(15), 2362–2368.

Mohamed, M.F., El-Gayyar, M., Shabayek, A.E.R., and Nassar, H. (2018). Data reduction in a cloud-based ami framework with service-replication. *Computers & Electrical Engineering*, 69, 212–223.

Rabileh, A.A., Bakar, K.A.A., Mohamed, R., and Mohamad, M. (2018). Enhanced buffer management policy and packet prioritization for wireless sensor network. *International Journal on Advanced Science, Engineering and Information Technology*, 8(4), 1770–1776.

Sah, D.K. and Amgoth, T. (2020). A novel efficient clustering protocol for energy harvesting in wireless sensor networks. *Wireless Networks*, 26, 4723–4737.

Shwe, H.Y., Gacanin, H., and Adachi, F. (2010). Multi-layer wsn with power efficient buffer management policy. In *2010 IEEE International Conference on Communication Systems*, 36–40. IEEE.

Ullah, I. and Youn, H.Y. (2019). A novel data aggregation scheme based on self-organized map for wsn. *The Journal of Supercomputing*, 75(7), 3975–3996.

Ullah, I. and Youn, H.Y. (2020). Efficient data aggregation with node clustering and extreme learning machine for wsn. *The Journal of Supercomputing*, 1–27.

Xu, X. and Zhang, G. (2017). A hybrid model for data prediction in real-world wireless sensor networks. *IEEE Communications Letters*.

Zainalie, S. and Yaghmaee, M.H. (2008). Cfl: A clustering algorithm for localization in wireless sensor networks. In *2008 International Symposium on Telecommunications*, 435–439. IEEE.